## REMARKS

Claims 1-66 were pending when the Office Action was mailed. Applicants herein amend claims 1, 23, 34, and 56 and do not cancel or present any new claims. Accordingly, claims 1-66 remain pending.

The following table reflects the rejections presented in the Office Action:

| Claims | Basis | References |
|---|---|---|
| 1-11, 15, 19-22, 34-44, 48, and 52-55 | 103(a) | Borkenhagen and Elliott |
| 12-14 and 45-47 | 103(a) | Borkenhagen, Elliott, and Alpert |
| 16 and 49 | 103(a) | Borkenhagen, Elliott, and Potash |
| 17, 18, 50, and 51 | 103(a) | Borkenhagen, Elliott, and Slingwine |
| 23-36 and 56-59 | 103(a) | Borkenhagen, Elliott, and AAPA |
| 27, 28, 60, and 61 | 103(a) | Borkenhagen, Elliott, AAPA, and Slingwine |
| 29-31 and 62-64 | 103(a) | Borkenhagen, Elliott, AAPA, and Alpert |
| 32, 33, 65, and 66 | 103(a) | Borkenhagen, Elliott, AAPA, and Shrote |

Applicants respectfully traverse these rejections. Nevertheless, applicants herein amend the claims to clarify the subject matter for which they seek protection.

Borkenhagen is directed to techniques for switching between threads in a multithreaded processor system. (Borkenhagen, Abstract). When a thread switch event occurs for an active thread, such as a latency event or an external interrupt signal, the state of the active thread can be stored and a waiting thread can be activated. (Borkenhagen, 6:22-41, 8:53-9:7). Elliott is directed to techniques for collecting statistical data from call detail records in a telephone network. (Elliott, 1:13-16). Elliott describes a receiver subsystem responsible for accepting connection requests and data from external clients. (Elliott, 9:3-6). The receiver subsystem includes an "Accept Requests" routine that listens for incoming connections to the system and starts a collector process to manage each connection. (Elliott, 9:20-27).

When the receiver subsystem is to be shut down, the Accept Requests routine is notified, which in turn notifies each of the collector processes. (Elliott, 9:38-44). If the collector processes do not shut down after a time-out period, the Accept Requests routine forces them to shut down and then shuts down itself.

In contrast, applicants' technology causes a task executing in a multithreaded environment to enter or exit a known state by causing each thread of the task to enter or exit the known state, the known state being an active state in which the thread is not making "productive use of the processor until an event occurs." (Specification, ¶ [0082]). The threads of the task have an active role in achieving the known state in a way that is not typical in other systems. Each time a task is to enter or exit a known state, applicants' technology designates a thread of the task to manage the state change. As each thread enters the known state, it notifies the designated thread. Once the designated thread has received notification from each thread, it notifies the operating system that the task is in the known state. When the task is to exit the known the state, the designated thread receives a notification from the operating system and notifies each of the other threads, which then begin executing instructions that were to be executed prior to entering the known state. In this manner, certain actions can be performed without corrupting the task because the task is in a known state that is active but the thread is not making productive use of the processor.

Claims 1, 23, 34, and 56 now recite "the known state being an active state in which the thread is not making productive use of the processor." As the Office Action points out, "Borkenhagen does not explicitly disclose of notifying a task to enter a known state." (Office Action, Pages 3, 10). To cure this deficiency, the Office Action relies on Elliott at 9:38-44 as disclosing "a method in which the threads executing are notified to enter a known state." (Office Action, Pages 3, 10). The relied-upon portion of Elliott describes shutting down a receiver process by notifying a routine so that the routine can notify each of a set of collector processes to shut down. If the collector processes are not stopped after a time-out period, the routine forces them to shut down and then stops

itself. Elliott's shut down process does not put each process into an "active state in which the process is not making productive use of the processor." Rather, Elliott's processes are completely terminated. Such a termination cannot be considered an "active" state. Borkenhagen and Elliott fail to teach or suggest notifying a task to enter or exit a known state wherein the known state is an active state in which each thread of the task is not making productive use of the processor. Accordingly, claims 1, 23, 34, and 56 are patentable over the cited references, as are their dependent claims 2-22, 24-33, 35-55, and 57-66.

In view of the above amendments and remarks, applicants believe the pending application is in condition for allowance and respectfully request reconsideration.

Please charge any deficiency in fees or credit any overpayment to our Deposit Account No. 50-0665, under Order No. 324758001US5 from which the undersigned is authorized to draw.

Dated: April 14, 2008

Respectfully submitted,

By 

Maurice J. Pirio
Registration No.: 33,273
PERKINS COIE LLP
P.O. Box 1247
Seattle, Washington 98111-1247
(206) 359-8000
(206) 359-7198 (Fax)
Attorney for Applicant